Toradex
Swiss. Embedded. Computing.

EMBEDDED LINUX CONFERENCE

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
EUROPE

Using OpenEmbedded inside containers? How and Why?

Presented by
Toradex

Toradex
Swiss. Embedded. Computing.

EMBEDDED LINUX CONFERENCE

THE LINUX FOUNDATION
OPEN SOURCE SUMMIT
EUROPE

Using OpenEmbedded inside containers? How and Why?

Presented by

Drew Moseley
Technical Solutions Architect
Toradex

What We'll Cover Today

Toradex
Swiss. Embedded. Computing.

- Goals
- Describe Containers
- Why Containers for Embedded
- Container tools and resources
- Building containers with OE
- meta-virtualization

Disclaimers

- I have only used Docker for any real work
- I am new to meta-virtualization
  - There is a lot in there I don't know
- This work is focused on functionality, not product-readiness and security
- Container technologies and tools are changing rapidly.

What We Do

Arm® System on Modules
Reliable
Long-term maintenance
Scalable
From stock

Torizon™

Production-ready software
Yocto-based Linux
Windows Embedded Compact
Development tools
Long-term maintenance

Toradex
Swiss. Embedded. Computing.

Ease-of-use
Support
Ecosystem

Survey

- Have you ever used containers?
- Have you ever used containers on an embedded device?
- Have you tried Containers on Torizon[1] or other commercial offering?
- How about directly through OpenEmbedded[2]?
- Are you a meta-virtualization layer contributor or maintainer?

[1] https://www.toradex.com/torizon
[2] https://www.openembedded.org/

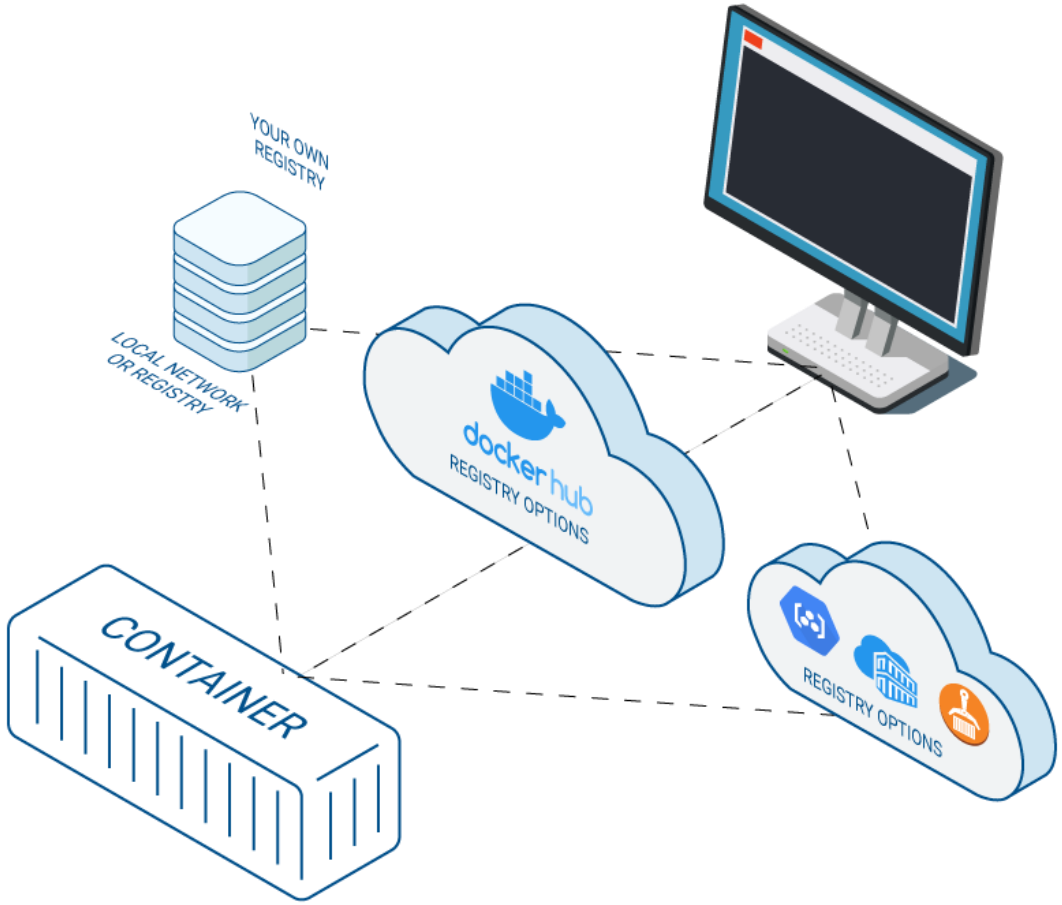
Containers

Container are not:

- Virtual Machines
- Universal applications
- Magic

Containers are:

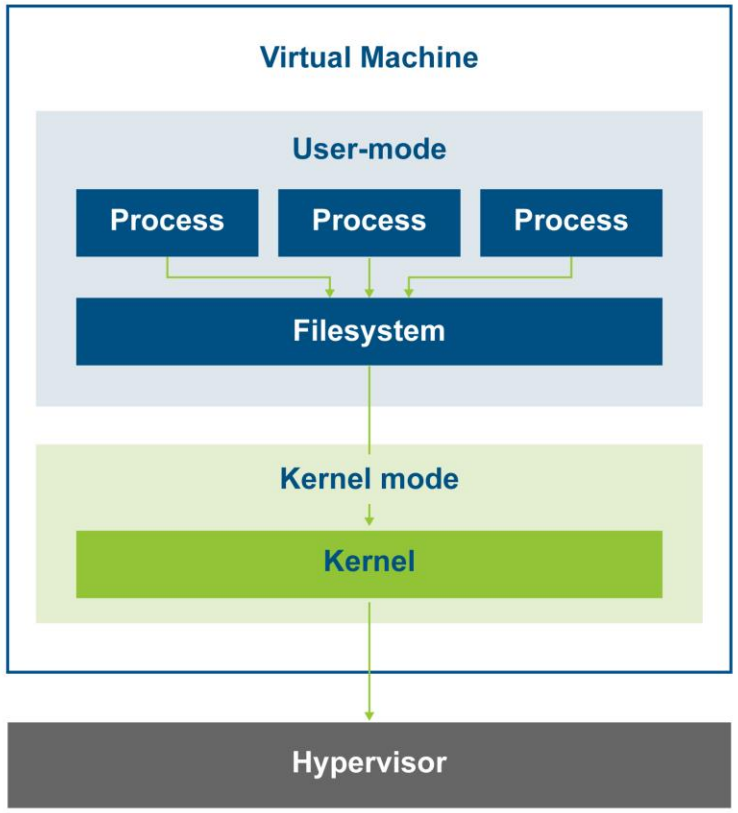
- A way to package software
- Including all dependencies
- A way to ensure a consistent runtime environment for your applications



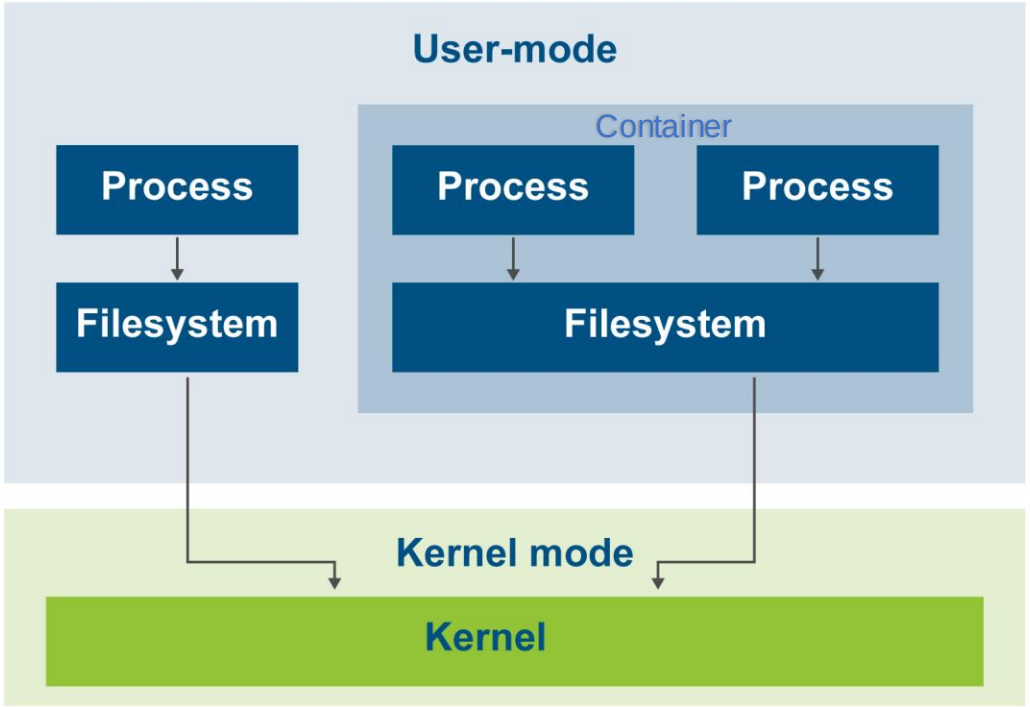
Containers vs VMs

Toradex
Swiss. Embedded. Computing.

Virtual Machine



Containers



Docker's Description

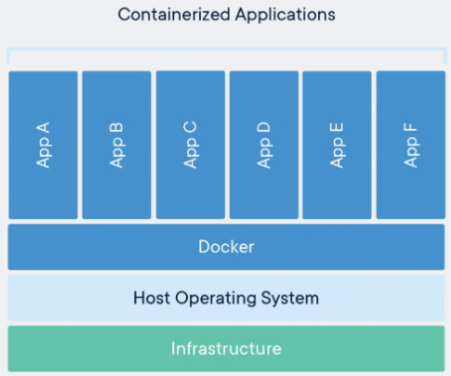
Use containers to Build, Share and Run your applications

Package Software into Standardized Units for Development, Shipment and Deployment

A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings.

Container images become containers at runtime and in the case of Docker containers – images become containers when they run on Docker Engine. Available for both Linux and Windows-based applications, containerized software will always run the same, regardless of the infrastructure. Containers isolate software from its environment and ensure that it works uniformly despite differences for instance between development and staging.

Docker containers that run on Docker Engine:



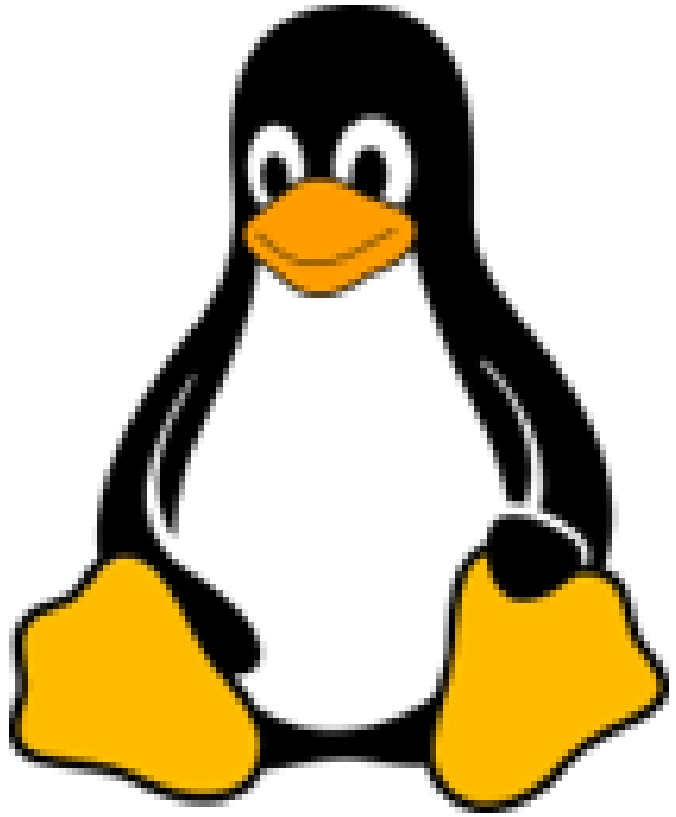
Source: https://www.docker.com/resources/what-container/

Container Building Blocks

Toradex
Swiss. Embedded. Computing.

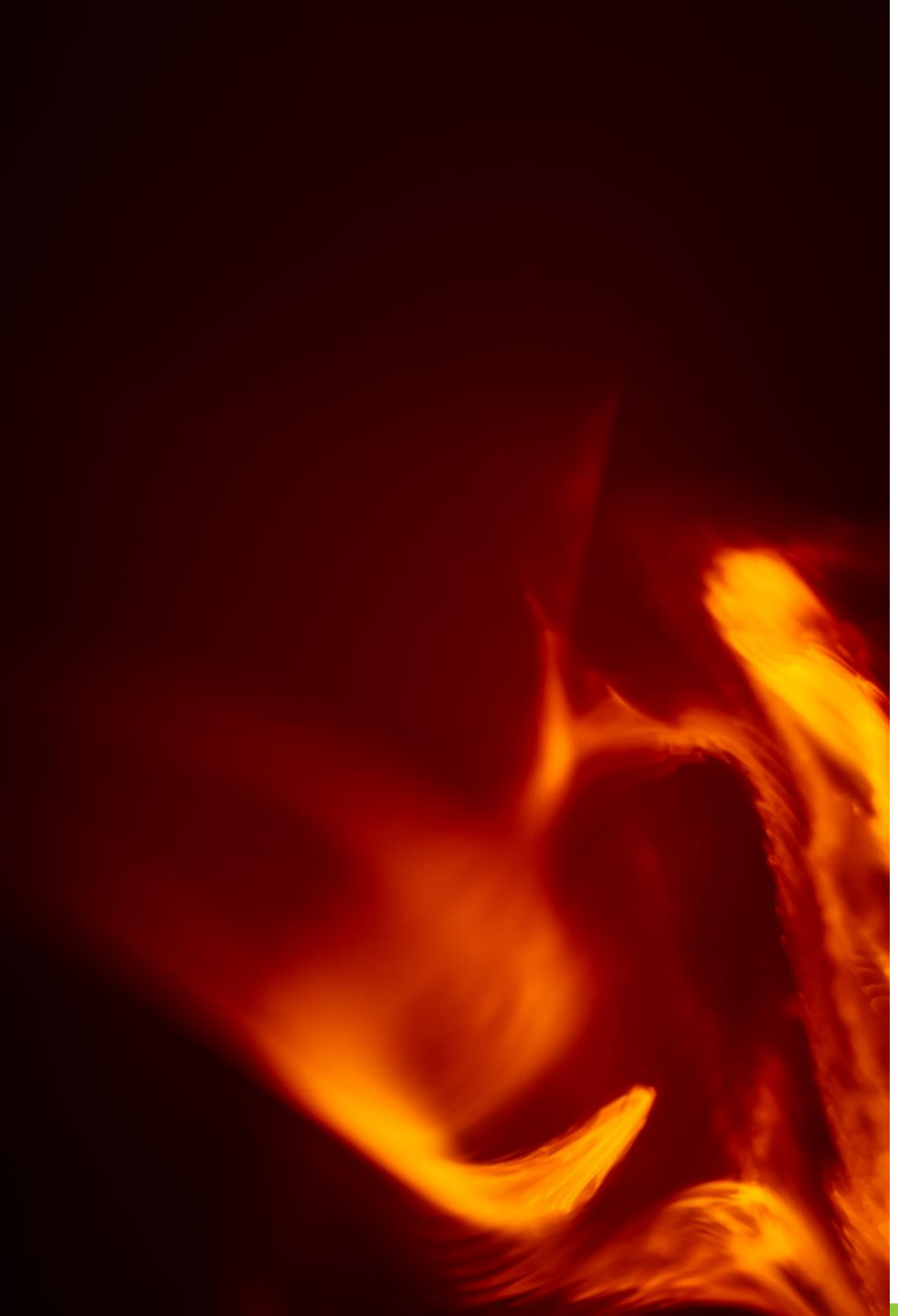
Standard Linux mechanisms:

- namespaces
- cgroups
- Networking components (ie bridges)



Implementations:

- Docker
- LXC
- Run
- systemd-nspawn



Toradex
Swiss. Embedded. Computing.

High-level Benefits of Containers (1/5)

- No dependency hell[1]

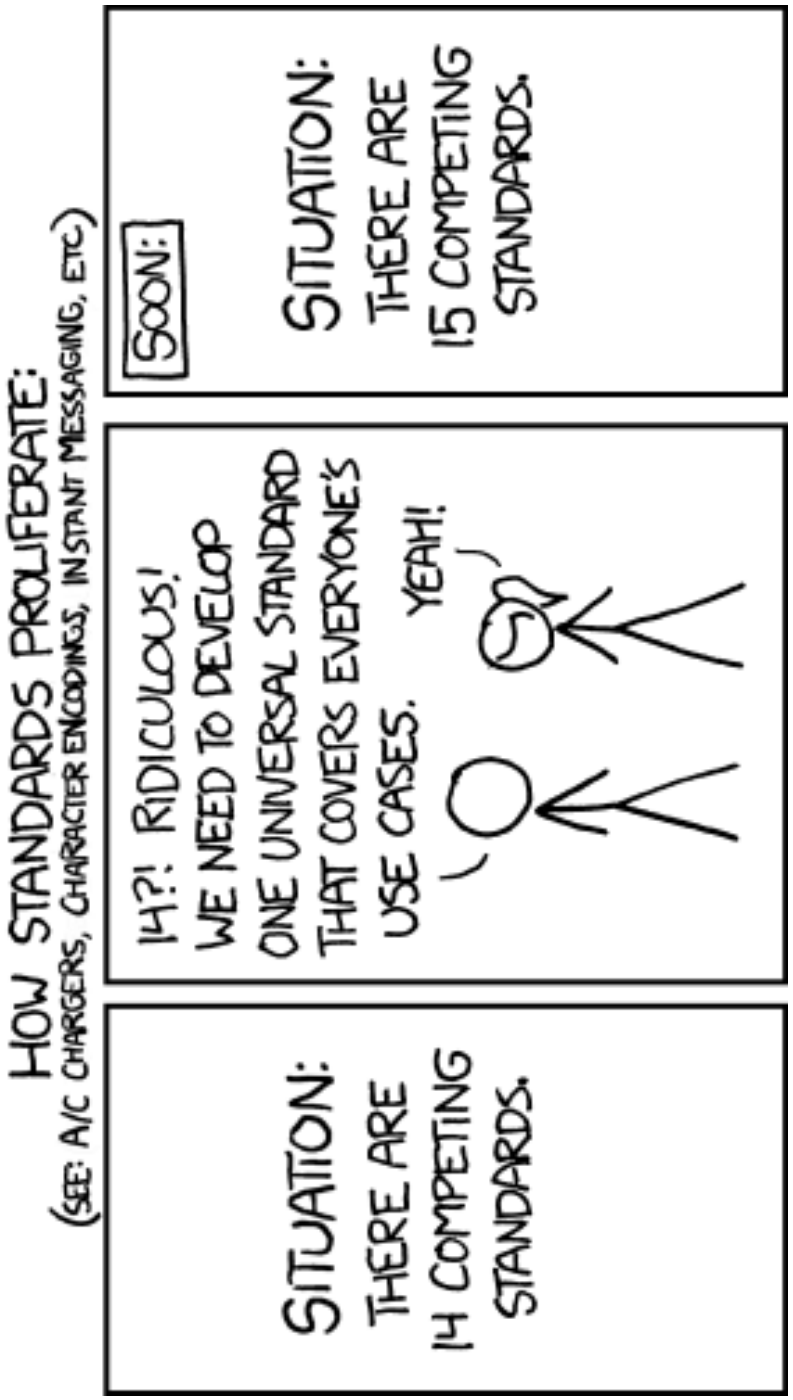
[1] https://about.sourcegraph.com/blog/nine-circles-of-dependency-hell

Toradex
Swiss. Embedded. Computing.

High-level Benefits of Containers (2/5)

- No dependency hell[1]
- Convenient package and delivery

[1] https://about.sourcegraph.com/blog/nine-circles-of-dependency-hell

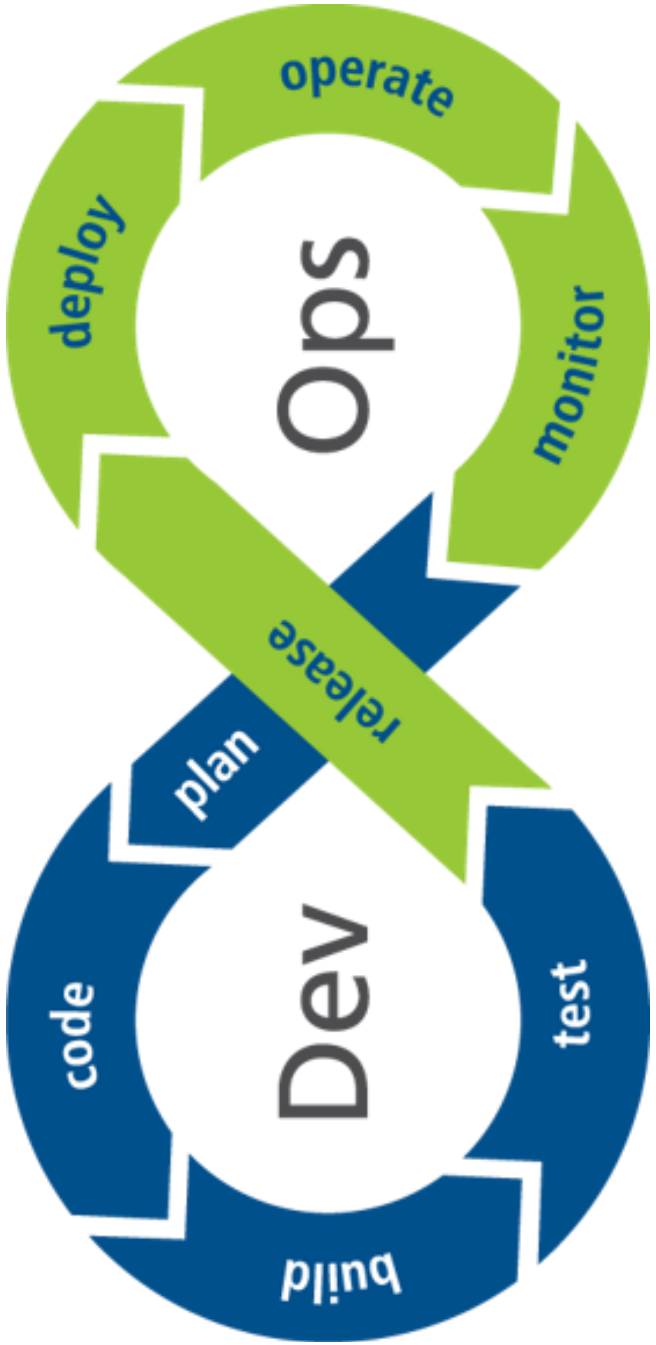


High-level Benefits of Containers (3/5)

- No dependency hell[1]
- Convenient package and delivery
- Standards based[2]

[1] https://about.sourcegraph.com/blog/nine-circles-of-dependency-hell

[2] https://xkcd.com/927

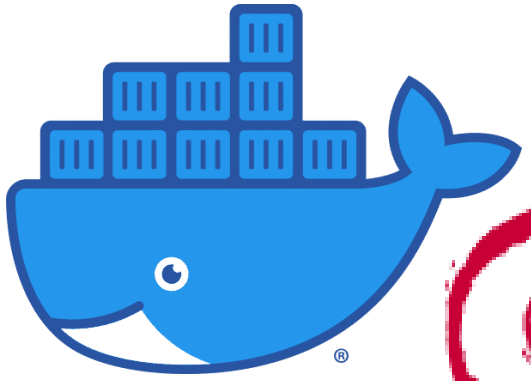


High-level Benefits of Containers (4/5)

- No dependency hell[1]
- Convenient package and delivery
- Standards based[2]
- Modern devops workflows

[1] https://about.sourcegraph.com/blog/nine-circles-of-dependency-hell

[2] https://xkcd.com/927



debian

NGINX®
Part of F5

Toradex
Swiss. Embedded. Computing.

High-level Benefits of Containers (5/5)

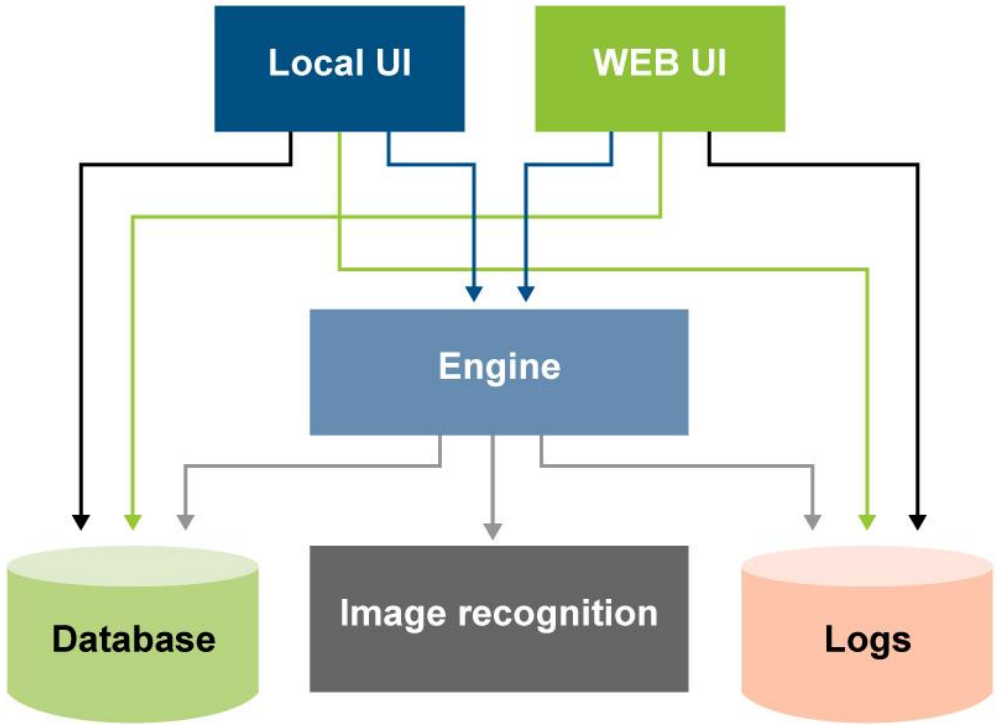
- No dependency hell[1]
- Convenient package and delivery
- Standards based[2]
- Modern devops workflow
- Lots of readily-available software

[1] https://about.sourcegraph.com/blog/nine-circles-of-dependency-hell

[2] https://xkcd.com/927

Technical Benefits of Containers

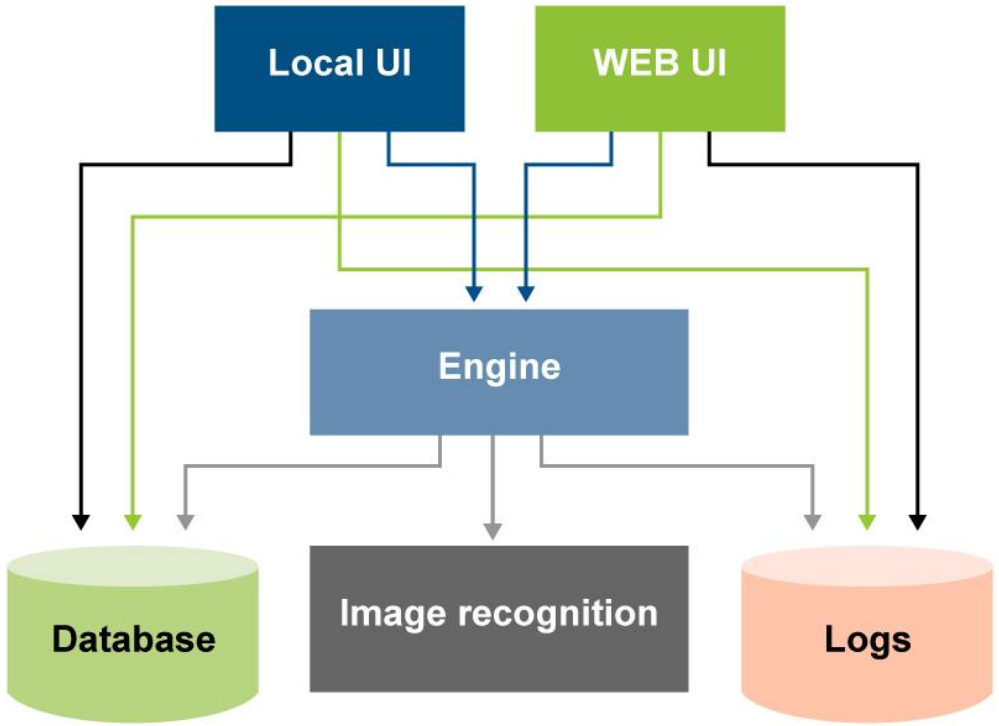
- Limit dependencies on OS-provided components
- Easily define multi-service architectures
- Individual service design can be simplified
- Can be based off familiar distros
- Isolation (CGroups, namespaces and chroot)



Technical Objections to Containers

Toradex
Swiss. Embedded. Computing.

- Runtime performance hit
- Increased storage/RAM footprint
  - › Mitigated by Docker layering mechanism
- New technology for engineering staff
- Overall design complexity



Containers for Embedded? Really??

Toradex
Swiss. Embedded. Computing.

Concerns:

- Startup time and runtime performance
- Defining multiple services (ie docker-compose)
- Hardware access
  - (typically) Everything is a file
  - Wayland
- Access to Host OS components

Torizon™

docker

openembedded

Containers for Embedded? Really??

Toradex
Swiss. Embedded. Computing.

Benefits:

- Devs can work in a familiar environment
- Unattended field updates are well supported
- Increased pool of app developers
- Apps easily ported from one machine to another (including desktops)
- Tighter control, isolation and limiting of hardware and software resources

Torizon™

docker

openembedded

Useful Container Resources and Tools

Toradex
Swiss. Embedded. Computing.

Open Container Initiative: https://opencontainers.org/

- "The mission of the Open Container Initiative (OCI) is to promote a set of common, minimal, open standards and specifications around container technology."
- Defines a format for encoding a container including JSON metadata

Podman: https://podman.io/

- "Podman is a daemonless container engine for developing, managing, and running OCI Containers on your Linux System"

Buildah: https://buildah.io/

- "a tool that facilitates building Open Container Initiative (OCI) container images"

Skopeo: https://github.com/containers/skopeo

- "a command line utility that performs various operations on container images and image repositories"

OpenEmbedded and Containers - Resources

With my thanks:

- Scott Murray at OpenEmbedded Workshop 2020:
  - https://www.youtube.com/watch?v=f6Yt3vSPMes
- Robert Berger at Yocto Project Summit 2020:
  - https://www.youtube.com/watch?v=n9NFRWzqdOA
- Bruce Ashfield at Yocto Project Virtual Summit, May 2021:
  - https://www.youtube.com/watch?v=HDSyILDGwfE

Creating Containers
The Docker Way

Toradex
Swiss. Embedded. Computing.

```
FROM crops/yocto:ubuntu-20.04-base
USER root
RUN echo "dash dash/sh boolean false" | debconf-set-selections
RUN DEBIAN_FRONTEND=noninteractive dpkg-reconfigure dash
RUN DEBIAN_FRONTEND=noninteractive apt update -y
RUN DEBIAN_FRONTEND=noninteractive apt install -y curl apt-utils ...
RUN DEBIAN_FRONTEND=noninteractive apt upgrade -y

RUN curl -s https://packagecloud.io/install/repositories/github/git-
lfs/script.deb.sh | sudo bash
RUN DEBIAN_FRONTEND=noninteractive apt-get install -y git-core git-lfs
RUN git lfs install
```

Toradex
Swiss. Embedded. Computing.

Image-container.bbclass

```
#
# Copyright OpenEmbedded Contributors
#
# SPDX-License-Identifier: MIT
#

ROOTFS_BOOTSTRAP_INSTALL = ""
IMAGE_TYPES_MASKED += "container"
IMAGE_TYPEDEP:container = "tar.bz2"

python __anonymous() {
    if "container" in d.getVar("IMAGE_FSTYPES") and \
       d.getVar("IMAGE_CONTAINER_NO_DUMMY") != "1" and \
       "linux-dummy" not in d.getVar("PREFERRED_PROVIDER_virtual/kernel"):
        msg = '"container" is in IMAGE_FSTYPES, but ' \
              'PREFERRED_PROVIDER_virtual/kernel is not "linux-dummy". ' \
              'Unless a particular kernel is needed, using linux-dummy will ' \
              'prevent a kernel from being built, which can reduce ' \
              'build times. If you don\'t want to use "linux-dummy", set ' \
              '"IMAGE_CONTAINER_NO_DUMMY" to "1".'

        # Raising skip recipe was Paul's clever idea. It causes the error to
        # only be shown for the recipes actually requested to build, rather
        # than bb.fatal which would appear for all recipes inheriting the
        # class.
        raise bb.parse.SkipRecipe(msg)
}
```

Build Containers with OpenEmbedded (1/3)

Toradex
Swiss. Embedded. Computing.

local.conf

```
IMAGE_FSTYPES='container'
PREFERRED_PROVIDER_virtual/kernel = 'linux-dummy'
```

Custom Image

```
SUMMARY = "Container image including python3"
DESCRIPTION = "Container image including python3"
LICENSE = "MIT"
LIC_FILES_CHKSUM = \
 "file://${COMMON_LICENSE_DIR}/MIT;md5=0835ade698e0bcf8506ecda2f7b4f302"
IMAGE_INSTALL = "busybox"
IMAGE_FEATURES = " "
inherit image
```

Build Containers with OpenEmbedded (2/3)

Toradex
Swiss. Embedded. Computing.

Build and deploy:

```
$ MACHINE=qemuarm64 bitbake busybox-container-image
$ scp tmp/deploy/images/qemuarm64/busybox-container-image-qemuarm64.tar.bz2 \
            torizon@apalis-imx8.lab.moseleynet.net:~/
```

Build Containers with OpenEmbedded (3/3)

Toradex
Swiss. Embedded. Computing.

Run container on board:

```
torizon@apalis-imx8-06805204:~$ docker import /tmp/busybox-container-image-qemuarm64.tar.bz2 minimalbusybox
sha256:54a3c8e340ffe4ce324c59266e04cdc6c14356b190248dc64db3a28c17f874fc
torizon@apalis-imx8-06805204:~$ docker images
REPOSITORY              TAG                 IMAGE ID            CREATED             SIZE
minimalbusybox          latest              54a3c8e340ff        3 seconds ago       7.21MB
torizon/chromium        2                   cb7ac4913734        2 weeks ago         609MB
torizon/weston-vivante  2                   eb17fa1e613f        2 weeks ago         424MB
orizon@apalis-imx8-06805204:~$ docker run --rm -it minimalbusybox /bin/sh
/ # ls /bin
ash             cp              false           ls              ping            sleep           usleep
base32          cpio            fgrep           mkdir           ping6           stat            vi
busybox         date            getopt          mknod           ps              stty            watch
```

Full Images

Toradex
Swiss. Embedded. Computing.

Build a full image;

```
$ MACHINE=qemuarm64 bitbake core-image-full-cmdline
$ docker import tmp/deploy/images/qemuarm64/core-image-full-cmdline-qemuarm64.tar.bz2 \
                      drewmoseley/core-image-full-cmdline-qemuarm64:latest
sha256:7f18e653c877acc42b832ae49ba06b52f76ec4b1a0e373d822907524b9de4858
$ docker push drewmoseley/core-image-full-cmdline-qemuarm64
Using default tag: latest
The push refers to repository [docker.io/drewmoseley/core-image-full-cmdline-qemuarm64]
8f6502b4b9f6: Pushed
latest: digest: sha256:4aa4f236dc327aa2a45c04043f3465a3541d3f317ceefdc3e5b2c274d2783cde size: 528
```

Run a full image;

```
torizon@apalis-imx8-06805204:~$ docker run --rm -it \
                      drewmoseley/core-image-full-cmdline-qemuarm64:latest /bin/sh
Unable to find image 'drewmoseley/core-image-full-cmdline-qemuarm64:latest' locally
latest: Pulling from drewmoseley/core-image-full-cmdline-qemuarm64
ed5fd0d8f642: Pull complete
Digest: sha256:4aa4f236dc327aa2a45c04043f3465a3541d3f317ceefdc3e5b2c274d2783cde
Status: Downloaded newer image for drewmoseley/core-image-full-cmdline-qemuarm64:latest
sh-5.1#
```

Size Reduction

Use musl (https://www.musl-libc.org/):

local.conf

```
TCLIBC="musl"
```

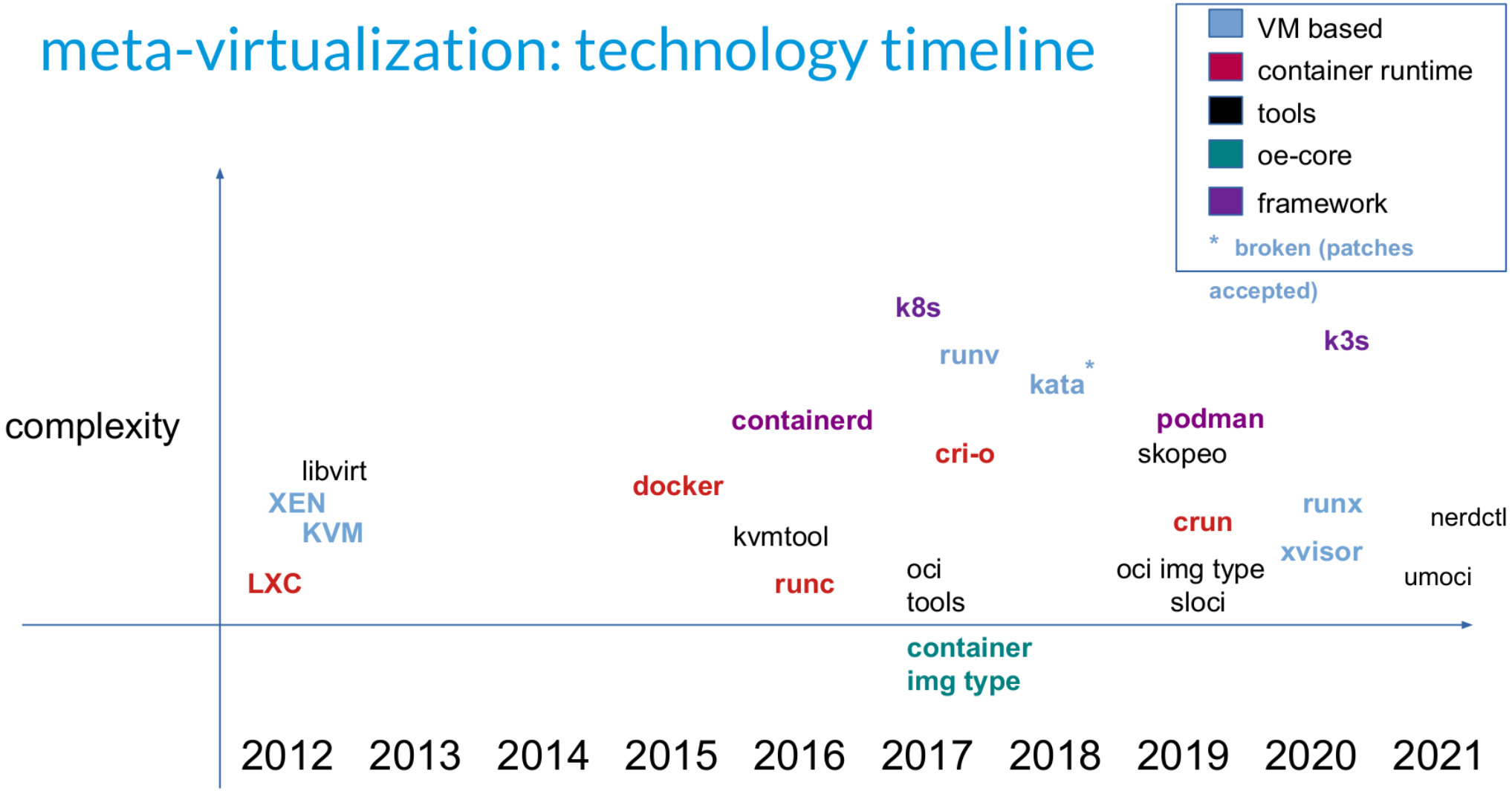
```
$ docker images | grep python-container
python-container-musl                 latest              2fca585515cd   5 seconds ago    56MB
python-container-glibc                latest              f0288a015645   59 seconds ago   58MB

$ docker images | grep busybox-container
busybox-container-musl                latest              8316cb14c47c   9 seconds ago    1.46MB
busybox-container-glibc               latest              81ace788a320   47 seconds ago   7.21MB
```

meta-virtualization

Toradex
Swiss. Embedded. Computing.

meta-virtualization: technology timeline



OpenEmbedded and Docker: (simple HowTo 1/3)

Toradex
Swiss. Embedded. Computing.

bblayers.conf

```
src/meta-openembedded/meta-oe \
src/meta-openembedded/meta-filesystems \
src/meta-openembedded/meta-python \
src/meta-openembedded/meta-networking \
src/meta-virtualization
```

local.conf

```
DISTRO_FEATURES:append = " virtualization "
IMAGE_INSTALL:append = " docker "
```

Options

```
PREFERRED_PROVIDER:virtual/docker = "docker-moby"
IMAGE_INSTALL:append = " podman "
```

OpenEmbedded and Docker: (simple HowTo 2/3)

Toradex
Swiss. Embedded. Computing.

```
root@qemux86-64:~# docker ps; ps -ef | grep dockerd; docker version
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
root         270       1  5 18:37 ?        00:00:05 /usr/bin/dockerd -H fd://
root         450     218  0 18:38 ttyS0    00:00:00 grep dockerd
Client:
Version:           20.10.12-ce
API version:       1.41
Go version:        go1.17.13
Git commit:        62eae52c2a
Built:             Fri Sep  9 14:24:18 2022
OS/Arch:           linux/amd64
Context:           default
Experimental:      true
```

OpenEmbedded and Docker: (simple HowTo 3/3)

Toradex
Swiss. Embedded. Computing.

```
root@qemux86-64:
CONTAINER ID   I
root         270
root         450
Client:
Version:
API version:
Go version:
Git commit:
Built:
OS/Arch:
Context:
Experimental:
```

```
Server:
Engine:
  Version:          20.10.12-ce
  API version:      1.41 (minimum version 1.12)
  Go version:       go1.17.13
  Git commit:       906f57ff5b-unsupported
  Built:            Tue Aug 17 12:11:07 2021
  OS/Arch:          linux/amd64
  Experimental:     false
containerd:
  Version:          v1.6.6-10-g4e92d8e7e.m
  GitCommit:        4e92d8e7e439530f5bb17e57a77481e9aa3da851.m
runc:
  Version:          1.1.2+dev
  GitCommit:        v1.1.2-9-Congb507e2da-dirty
docker-init:
  Version:          0.19.0
  GitCommit:        b9f42a0-dirty
```

Why do all this?

- Reproducibility and repeatability of builds
- Can be completely self-hosted
- Source archival and BOM maintenance
- License tracking and compliance
- Better visibility into container contents
- Leverage OpenEmbedded deep configurability

Future Work

- Setup a container repository
- Generate a usable set of containers
- Test end-to-end in a production system
- Use proper MACHINE config for compiler tuning
- Inspect all packages included in image for further reduction
- Investigate multiconfig setup
- Figure out build error with image type "oci"
- Learn more about meta-virtualization options and other container related tools

Toradex
Swiss. Embedded. Computing.

Q&A

Toradex
Swiss. Embedded. Computing.

EMBEDDED LINUX CONFERENCE

THE LINUX FOUNDATION OPEN SOURCE SUMMIT EUROPE

THANK YOU
FOR YOUR INTEREST

www.**toradex**.com
developer.toradex.com
community.toradex.com
labs.toradex.com

Torizon™